AD_____

AWARD NUMBER DAMD17-98-1-8061

TITLE: Application of Information Theory to Improve Computer-Aided Diagnosis Systems

PRINCIPAL INVESTIGATOR: Paul Sajda, Ph.D.

CONTRACTING ORGANIZATION: Sarnoff Corporation
Princeton, New Jersey 08543-5300

REPORT DATE: August 1999

TYPE OF REPORT: Annual

PREPARED FOR: U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1999	Annual (1 Jul 98 - 1 Jul 99)

4. TITLE AND SUBTITLE
Application of Information Theory to Improve Computer-Aided Diagnosis Systems

5. FUNDING NUMBERS
DAMD17-98-1-8061

6. AUTHOR(S)
Paul Sajda, Ph.D.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Sarnoff Corporation
Princeton, New Jersey 08543-5300

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Mammographic Computer-aided diagnosis (CAD) systems are an approach for low-cost double reading. Though results to date have been promising, current systems often suffer from unacceptably high false positive rates. Improved methods are needed for optimally setting the system parameters, particularly in the case of statistical models and neural networks which are common elements of most CAD systems. This research project looks to apply principles from information theory to build statistical models for CAD systems. Specifically, we develop a framework for building hierarchical pattern recognizers based on the minimum description length principle (MDL). Under the first year of this project we have developed a framework for building generative hierarchical image probability (HIP) models. Since the HIP framework is a generative model which directly models the probability of the image given the image class, it is well-suited to compression and application of MDL. We have started building HIP architectures using the MDL selection criteria. For example we have used predictive MDL (pMDL) to select the number of segmentation labels at each level of the pyramid. Finally, under our first year's effort, we have also expanded our hierarchical modeling framework to include both microcalcification and mass detection.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Breast Cancer Computer-aided Diagnosis, Mammography, Model Selection, Minimum Description Length Principle, Hierarchical Image Probability, Hierarchical Pyramid Neural Network	22
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18 298-102

USAPPC V1.00

FOREWORD

Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the U.S. Army.

____ Where copyrighted material is quoted, permission has been obtained to use such material.

____ Where material from documents designated for limited distribution is quoted, permission has been obtained to use the material.

____ Citations of commercial organizations and trade names in this report do not constitute an official Department of Army endorsement or approval of the products or services of these organizations.

____ In conducting research using animals, the investigator(s) adhered to the "Guide for the Care and Use of Laboratory Animals," prepared by the Committee on Care and use of Laboratory Animals of the Institute of Laboratory Resources, national Research Council (NIH Publication No. 86-23, Revised 1985).

____ For the protection of human subjects, the investigator(s) adhered to policies of applicable Federal Law 45 CFR 46.

____ In conducting research utilizing recombinant DNA technology, the investigator(s) adhered to current guidelines promulgated by the National Institutes of Health.

____ In the conduct of research utilizing recombinant DNA, the investigator(s) adhered to the NIH Guidelines for Research Involving Recombinant DNA Molecules.

____ In the conduct of research involving hazardous organisms, the investigator(s) adhered to the CDC-NIH Guide for Biosafety in Microbiological and Biomedical Laboratories.

Paul Sydo 7/28/99

PI - Signature Date

Table of Contents

Applications of Information Theory to Improve Computer-Aided Diagnosis Systems

Year 1 Progress Report

Paul Sajda, Clay Spence and Lucas Parra
Sarnoff Corporation
CN5300
Princeton, NJ 08543-5300
{psajda, cspence, lparra}@sarnoff.com

July 28, 1999

1 Introduction

Computer-aided diagnosis (CAD) systems for mammography, under development for more than 10 years, are an approach for low-cost double-reading with potential to improve the detection of breast cancer. Though results to date have been promising, current systems often suffer from unacceptably high false positive rates and lower than expected sensitivity and specificity when evaluated on new data. Improved methods are needed for optimally setting the system parameters, particularly in the case of statistical models and neural networks which are a common element of most CAD systems. This research project looks to apply principles from information theory to build improved statistical models for CAD systems. Specifically, we develop a framework for building hierarchical pattern recognizers based on the minimum description length principle (MDL) pioneered by Rissanen [9]. Under the first year of this project we have developed a framework for building generative hierarchical image probability (HIP) models. Since the HIP framework is a generative model which directly models the probability of the image given the image class, it is well-suited to compression and thus application of MDL. We have started building HIP architectures using the MDL selection criteria. For example we have used predictive MDL (pMDL) to select the number of segmentation labels at each level of the pyramid. Finally, under our first year's effort, we have also expanded our hierarchical modeling framework to include both microcalcification and mass detection. Years two and three of our effort will focus on a more thorough application of MDL to our HIP architecture and a more rigorous analysis of the performance of HIP when integrated into the University of Chicago's (UofC) CAD systems for microcalcification and mass detection.

2 Body

The following are the three primary tasks under the first year of the project.

1. Apply and evaluate the utility of our hierarchical pyramid neural network (HPNN) architecture for improving mass detection in a CAD system.
2. Develop basic MDL framework within context of building models for CAD applications–development of HIP framework.
3. Apply MDL framework to select optimal number of nodes (labels) for statistical (HIP) models.

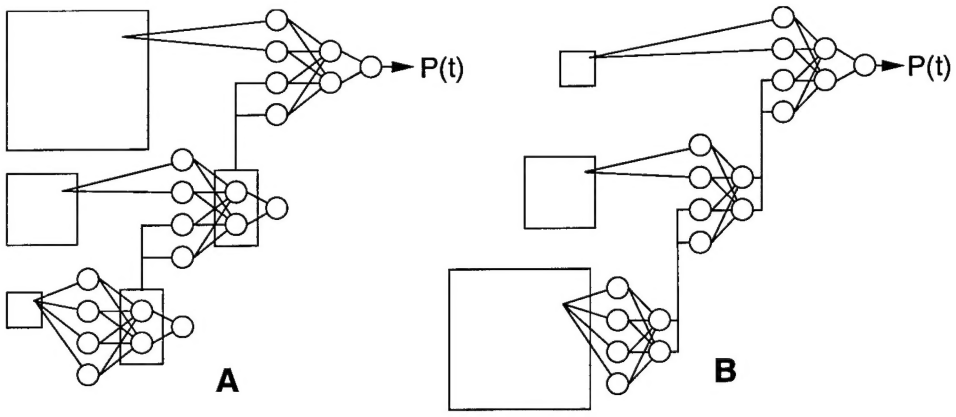

Figure 1: Hierarchical pyramid/neural network architectures for (A) detecting microcalcifications and (B) detecting masses. In (A) context is propagated from low to high resolution via the hidden units of low resolution networks. In (B) small-scale detail information is propagated from high to low resolution. In both cases the output of the last integration network is an estimate of the probability that a target is present.

In the following sections we describe in detail our progress in accomplishing these tasks, including the theoretical and experimental results obtained thus far.

2.1 Mass detection using the HPNN

Our previous work [11] presented a coarse-to-fine hierarchical pyramid/neural network (*HPNN*) architecture that combines multi-scale image processing techniques with neural networks to detect microcalcifications in digital/digitized mammograms (see figure 1A). To *search* an image we apply the network at a position and use its output as an estimate of the probability that a microcalcification is present. We then repeat this at each position in the image. In the coarse-to-fine HPNN, the hidden units of networks operating at low resolution or coarse scale learn associated *context* information, since the targets themselves are difficult to detect at low resolution. The context is then passed to networks searching at higher resolution. The use of context can significantly improve detection performance since microcalcifications have few distinguishing features. In the HPNN, each of the networks receives information directly from only a small part of several feature images and so the networks can be relatively simple. The network at the highest resolution integrates the contextual information learned at coarser resolutions to detect the object of interest.

Under this project, we have extended the HPNN architecture by considering the implications of inverting the information flow in the coarse-to-fine architecture. This fine-to-coarse HPNN has networks extracting detail structure at fine resolutions of the image and then passing this detail information to networks operating at coarser scales (see figure 1B). For many types of objects, for example mammographic masses, information about the fine structure is important for discriminating between different classes. The fine-to-coarse HPNN is therefore a natural architecture for exploiting fine detail information for detecting extended objects.

2.1.1 Mass detection experimental results

Radiologists often distinguish malignant from benign masses based on the detailed shape of the mass border and the presence of spicules alone the border. Thus to integrate this high resolution information to detect malignant masses, which are extended objects, we apply the fine-to-coarse HPNN of figure 1B.

As for microcalcifications [11], we apply the HPNN as a post-processor, but here it processes the output of the mass-detection component of UofC CAD system. The data in our study consists of 72 positive and 100 negative ROIs. These are 256-by-256 pixels and are sampled at 200 micron resolution. Half the data was used for training and half

Sensitivity	Fine-to-Coarse HPNN Specificity Mass
100 %	51 %
95 %	57 %
90 %	67 %
80 %	79 %

Table 1: Detector Specificity (% reduction in false positive rate of UofC CAD system).

for testing.

At each level of the fine-to-coarse HPNN several hidden units process the feature images. The outputs of each unit at all of the positions in an image make up a new feature image. This is reduced in resolution by the usual pyramid blur-and-subsample operation to make an input feature image for the network units at the next lower resolution. We trained the entire fine-to-coarse HPNN as one network instead of training a network for each level, one level at a time. This training is quite straightforward. Back-propagating error through the network units is the same as in conventional networks. We must also back-propagate through the pyramid reduction operation, but this is linear and therefore quite simple. In addition we use the same UOP error function used in our previous work to to train the coarse-to-fine architecture [12]. The rationale for this application of the UOP error function is that the truth data specifies the location of the center of the mass at the highest resolution. However, because of the sub-sampling the center cannot be unambiguously assigned to a particular pixel at low resolution.

The features input to the fine-to-coarse HPNN are filtered versions of the image, with filter kernels given by $\psi_{q,p}(r,\theta) = \left(\frac{q!}{\pi(q+|p|)!}\right)^{1/2} r^{|p|} e^{-r^2/2} L_q^{|p|}(r^2) e^{ip\phi}$ in polar coordinates, with $(q,p) \in \{(0,1),(1,0),(0,2)\}$. These are combinations of derivatives of Gaussians, and can be written as combinations of separable filter kernels (products of purely horizontal and vertical filters), so they can be computed at relatively low cost. They are also easy to steer, since this is just multiplication by a complex phase factor. We steered these in the radial and tangential directions relative to the tentative mass centers, and used the real and imaginary parts and their squares and products as features. The center coordinates of the are generated by the earlier stages of the CAD system. These features were extracted at each level of the Gaussian pyramid representation of the mass ROI, and used as inputs only to the network units at the same level.

The fine-to-coarse HPNN is quite similar to the convolution network proposed by Le Cun, et al [5], however with a few notable differences. The fine-to-coarse HPNN receives as inputs preset features extracted from the image (in this case radial and tangential gradients) at each resolution, compared to the convolution network, whose inputs are the original pixel values at the highest resolution. Secondly, in the fine-to-coarse HPNN, the inputs to a hidden unit at a particular position are the pixel values at that position in each of the feature images, one pixel value per feature image. Thus the HPNN's hidden units do not learn linear filters, except as linear combinations of the filters used to form the features. Finally the fine-to-coarse HPNN is trained using the UOP error function, which is not used in the Le Cun network.

Currently our best performing fine-to-coarse HPNN system for mass detection has two hidden units per pyramid level. This gives an ROC area of $A_z = 0.85$ and eliminates 51 % of the false-positives without any loss in sensitivity.

2.2 A Hierarchical Image Probability framework for MDL

Many approaches to object recognition in images, including those used in mammographic CAD, estimate $\Pr(\text{class} \mid \text{image})$, the probability that, given an image, it is an image of an object of a particular class. This in fact is the approach that has been used in our HPNN framework. By contrast, a model of the probability distribution of images, $\Pr(\text{image} \mid \text{class})$, has many attractive features. We could use this for object recognition in the usual way by training a distribution for each object class and using Bayes' rule to get $\Pr(\text{class} \mid \text{image}) = \Pr(\text{image} \mid \text{class}) \Pr(\text{class}) / \Pr(\text{image})$. Since we would have $\Pr(\text{image} \mid \text{class})$, we could attempt to detect unusual images and reject them rather than trust the

classifier; something that is not possible with models of $\Pr(\text{class} \mid \text{image})$. Since were are "generating" a model of the distribution of the data, this type of model is typically called a *generative* model.

Though our results for the application of the HPNN to mammographic CAD have been promising, the HPNN is not a good framework for applying MDL techniques for model selection. Since the HPNN estimates $\Pr(\text{class} \mid \text{image})$ there is one bit of information per example (i.e. the region of interest either has a mass or it does not). Thus one would require many examples to build up a sufficient number of samples for getting a robust MDL cost. Alternatively, if one estimates $\Pr(\text{image} \mid \text{class})$, one has an entire distribution (the distribution of the image) to compute an MDL cost for a given example. Since in CAD we typically work with images that are 256-by-256 this dramatically increases the number of bits available for computing the description length. This is important given that many MDL techniques are only valid asymptotically [10]–i.e. large amount of data. In addition, it is more intuitive to think about MDL in terms of "compressing data". One can think about MDL encoding the compactness of the modeled distribution together with the likelihood of the data under the modeled distribution.

Though the HPNN is not ideally suited for the application of MDL, it does have some attractive features. Most importantly, the HPNN is a framework for learning and integrating multi-resolution information for object classification. For instance, the HPNN is able to improve microcalcification detection performance for the University of Chicago CAD system because it can exploit low resolution contextual information, such as the location of blood vessels and the ductual system [11]. Thus a generative modeling framework should also take advantage of multi-resolution information for exploiting contextual information.

In the following section we briefly describe previous work in modeling the probability distributions of images. We then describe the new framework we have developed, which we call hierarchical image probabilities (HIP). We present the theory behind the framework and then our initial results in applying the HIP model to mammographic mass detection.

2.3 Previous work in modeling image probability distributions

Many image analysis algorithms use probability concepts, but few treat the distribution of images. Zhu, Wu and Mumford [14] do this by computing the maximum entropy distribution given a set of statistics for some features. This works well for textures but it is not clear how well it will model the appearance of more structured objects. In addition, with their approach it is easy to compute the probability of an image but harder to sample from the distribution, i.e., generate new artificial images. The ability to sample is necessary for many image analysis applications, e.g., compression.

There are several algorithms for modeling the distributions of features extracted from the image, instead of the image itself. The Markov Random Field (*MRF*) models are an example of this line of development; see, e.g., [6, 4]. Unfortunately they tend to be very expensive computationally. Because it is not an image distribution it only applies to some image analysis tasks, such as texture classification, that do not require sampling.

In De Bonet and Viola's flexible histogram approach [2, 1], features are extracted at multiple image scales, and the resulting feature vectors are treated as a set of independent samples drawn from a distribution. They then model this distribution of feature vectors with Parzen windows. The flexible histogram approach has given good results, but the feature vectors from neighboring pixels are treated as independent when in fact they share exactly the same components from lower-resolutions. To fix this one might build a model in which the features at one pixel of one pyramid level condition the features at each of several child pixels at the next higher-resolution pyramid level.

The multi-scale stochastic process (*MSP*) methods do exactly that. Luettgen and Willsky [8], for example, applied a scale-space auto-regression (AR) model to texture discrimination. They use a quadtree or quadtree-like organization of the pixels in an image pyramid, and model the features in the pyramid as a stochastic process from coarse-to-fine levels along the tree. The variables in the process are hidden, and the observations are sums of these hidden variables plus noise.

However, the MSP model distributions are Gaussian, i.e., the joint distribution of all of the variables is a Gaussian distribution. This is clearly not the case in natural images, such as mammograms.

Buccigrossi and Simoncelli [3], for example, have found that the distributions of some features have high kurtosis,

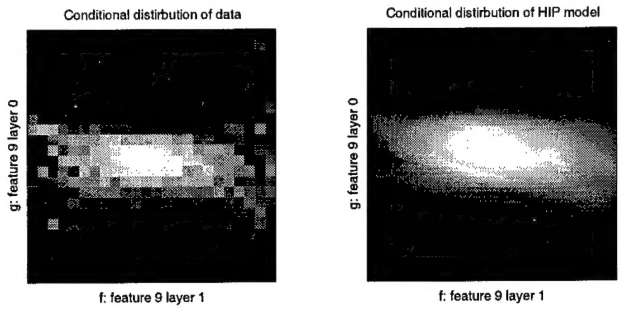

Figure 2: Empirical and HIP estimates of the distribution of a feature $g_l(x)$ conditioned on its parent feature $f_{l+1}(x)$.

and that the distribution of one feature conditioned on a neighboring feature has a "bow-tie" shape, which cannot follow from a Gaussian joint distribution. We have obtained similar results using our HIP model (Figure 2). The MSP approach also models the probability of the observations on the tree, not the probability of the image. Because neither the flexible histogram nor MSP approaches model the image distribution neither are well-suited for application of MDL.

All of these methods seem well-suited for modeling texture, but it is unclear how we might build the models to capture the appearance of more structured objects or objects which are hybrid in nature (e.g. which include both structure and texture), such as mammographic masses. We can argue that the presence of objects in images can make local conditioning like that of the flexible histogram and MSP approaches inappropriate. Objects in the world cause correlations and non-local dependencies in images across different resolutions. For example, the presence of a particular object might cause a certain kind of texture to be visible at some resolution. Usually the local image structure at lower resolutions by itself will not contain enough information to infer the object's presence, but the entire image at lower resolutions might. Therefore the probability that a texture is present will depend on a large region in the lower-resolution image.

Similarly, objects create long-range spatial dependencies at a given resolution. For example, an object class might result in a kind of texture across a large area of the image. If an object of this class is always present, we would know that the texture is present. But if such objects are not always present and cannot be inferred from lower-resolution information, knowing that the texture is present at one location tells us that it is present elsewhere.

These considerations imply that the assumptions of the flexible histogram and MSP approaches are not correct. The features at one resolution and one location depend on lower-resolution image information over a large area of the image, and even given that information they depend on the features at other locations at that resolution.

Thus the current state-of-the-art in image probability modeling is deficient. The best models we know of either cannot model complex long-range structure (at least it is not obvious that they can), or they cannot model the distribution of images, and thus cannot fully take advantage of model selection using MDL.

2.3.1 The Theory behind HIP

We have developed a model for probability distributions of images, in which we try to move beyond texture modeling and toward an approach which could model more structured objects, such as mammographic masses and microcalcifications. This *Hierarchical Image Probability* or *HIP* model is similar to a hidden Markov model on a tree, and can be learned with the EM algorithm. This section presents the details of the model.

Coarse-to-fine factoring of image distributions Our goal will be to write the image distribution in a form similar to $\Pr(I) \sim \Pr(\mathbf{F}_0 \mid \mathbf{F}_1) \Pr(\mathbf{F}_1 \mid \mathbf{F}_2) \ldots$, where $\mathbf{F}_l$ is the set of feature images at pyramid level l. We expect that the short-range dependencies can be captured by the model's distribution of individual feature vectors, while the long-range dependencies can be captured somehow at low resolution. The large-scale structures affect finer scales by the

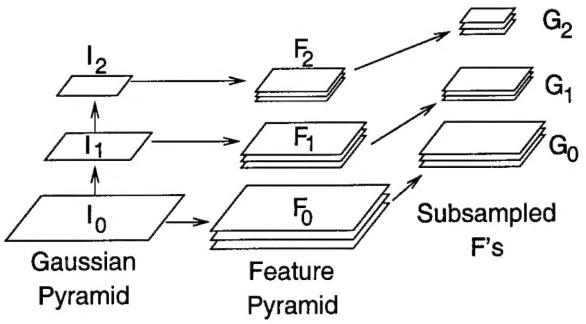

Figure 3: Pyramids and feature notation.

conditioning.

In fact we can prove that a coarse-to-fine factoring like this is correct. From an image I we build a Gaussian pyramid (repeatedly blur-and-subsample, with a Gaussian filter). Call the l-th level I_l, e.g., the original image is I_0 (Figure 3). From each Gaussian level I_l we extract some set of feature images $\mathbf{F}_l$. Sub-sample these to get feature images $\mathbf{G}_l$. Note that the images in $\mathbf{G}_l$ have the same dimensions as I_{l+1}. We denote by $\tilde{\mathbf{G}}_l$ the set of images containing I_{l+1} and the images in $\mathbf{G}_l$. We further denote the mapping from I_l to $\tilde{\mathbf{G}}_l$ by $\tilde{\mathcal{G}}_l$.

Suppose now that $\tilde{\mathcal{G}}_0 : I_0 \mapsto \tilde{\mathbf{G}}_0$ is invertible. Then we can think of $\tilde{\mathcal{G}}_0$ as a change of variables. If we have a distribution on a space, its expressions in two different coordinate systems are related by multiplying by the Jacobian. In this case we get $\Pr(I_0) = |\tilde{\mathcal{G}}_0| \Pr(\tilde{\mathbf{G}}_0)$. Since $\tilde{\mathbf{G}}_0 = (\mathbf{G}_0, I_1)$, we can factor $\Pr(\tilde{\mathbf{G}}_0)$ to get $\Pr(I_0) = |\tilde{\mathcal{G}}_0| \Pr(\mathbf{G}_0 \mid I_1) \Pr(I_1)$. If $\tilde{\mathcal{G}}_l$ is invertible for all $l \in \{0, \ldots, L-1\}$ then we can simply repeat this change of variable and factoring procedure to get

$$\Pr(I) = \left[\prod_{l=0}^{L-1} |\tilde{\mathcal{G}}_l| \Pr(\mathbf{G}_l \mid I_{l+1}) \right] \Pr(I_L) \tag{1}$$

Hidden variables model non-local dependencies For the sake of tractability we want to factor $\Pr(\mathbf{G}_l \mid I_{l+1})$ over positions. We might do this by replacing I_{l+1} with $\mathbf{G}_{l+1}$ and I_{l+2}, since together they carry the same information as I_{l+1}. In order to factor we would rather condition on images that are the same size as $\mathbf{G}_l$, so we replace $\mathbf{G}_{l+1}$ with $\mathbf{F}_{l+1}$. This is valid since both are derived from I_{l+1}, i.e., I_{l+1}, $(\mathbf{G}_{l+1}, I_{l+2})$, and $(\mathbf{F}_{l+1}, I_{l+2})$ all carry the same information. If we then drop the conditioning on I_{l+2}, since it is smaller than $\mathbf{G}_l$, we get $\Pr(\mathbf{G}_l \mid \mathbf{F}_{l+1})$. (We might reason that I_{l+2} carries only the local average brightness anyway, and perhaps this isn't important for $\mathbf{G}_l$.) We could now factor this over positions to get

$$\Pr(I) \sim \prod_l \prod_{x \in I_{l+1}} \Pr\big(\mathbf{g}_l(x) \mid \mathbf{f}_{l+1}(x)\big),$$

where $\mathbf{g}_l(x)$ and $\mathbf{f}_{l+1}(x)$ are the feature vectors at position x.

The dependence of $\mathbf{g}_l$ on $\mathbf{f}_{l+1}$ expresses the persistence of image structures across scale, e.g., an edge is usually detectable as such in several neighboring pyramid levels. The flexible histogram and MSP methods share this structure.

While it may be plausible that $\mathbf{f}_{l+1}(x)$ has a strong influence on $\mathbf{g}_l(x)$, we have argued above that this factorization and conditioning is not enough to capture some properties of real images, namely the dependence of a feature on large regions of the lower-resolution image, and the dependence between features at distant locations in the same resolution.

To represent the non-local information that is not captured by local features, we introduce hidden variables. They should also constrain the variability of features at the next finer scale, giving a more compact distribution. Denoting the hidden variables collectively by A, we assume that conditioning on A allows the distributions over feature vectors

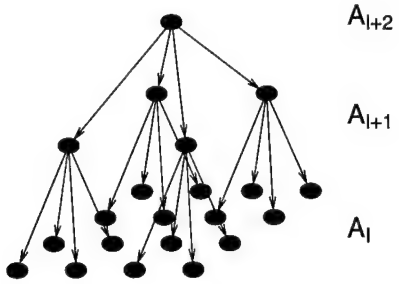

Figure 4: Tree structure of the conditional dependency between hidden variables in the HIP model. With sub-sampling by two, this is sometimes called a quadtree structure.

to factor. In general, the distribution over images becomes

$$\Pr(I) \propto \sum_{A} \left\{ \prod_{l=0}^{L-1} \prod_{x \in I_{l+1}} \Pr\big(\mathbf{g}_l(x) \,\big|\, \mathbf{f}_{l+1}(x), A\big) \right\} \Pr(I_L \,|\, A) \Pr(A). \tag{2}$$

Any distribution of images can be written in this way, at least in principle, since A, $\Pr(A)$, and the dependence of the image features on A can have arbitrarily complex structure. So we need to be more specific. In particular we would like to preserve the conditioning of higher-resolution information on coarser-resolution information, and the ability to factor over positions.

As a first model we have chosen the following structure for our HIP model:[1]

$$\Pr(I) \propto \sum_{A_0,\ldots,A_{L-1}} \prod_{l=0}^{L} \prod_{x \in I_{l+1}} \Big[\Pr\big(\mathbf{g}_l(x) \,\big|\, \mathbf{f}_{l+1}(x), a_l(x)\big) \Pr\big(a_l(x) \,\big|\, a_{l+1}(x)\big) \Big]. \tag{3}$$

To each position x at each level l we attach a hidden discrete index or label $a_l(x)$. The resulting label image A_l for level l has the same dimensions as the images in $\tilde{\mathbf{G}}_l$.

Since $a_l(x)$ codes non-local information we can think of the labels A_l as a segmentation or classification at the l-th pyramid level. By conditioning $a_l(x)$ on $a_{l+1}(x)$, we mean that $a_l(x)$ is conditioned on a_{l+1} at the *parent* pixel of x. This parent-child relationship follows from the sub-sampling operation. For example, if we sub-sample by two in each direction to get $\mathbf{G}_l$ from $\mathbf{F}_l$, we condition the variable a_l at (x, y) in level l on a_{l+1} at location $(\lfloor x/2 \rfloor, \lfloor y/2 \rfloor)$ in level $l+1$ (Figure 4). This gives the dependency graph of the hidden variables a tree structure. Such a probabilistic tree of discrete variables is a particular kind of belief network. By conditioning child labels on their parents information propagates though the layers to other areas of the image while accumulating information along the way.

2.3.2 Training the model with an EM algorithm

The EM algorithm can be applied here in a straight-forward manner. First the expectations over the hidden variables of the log-likelihood are computed for a given set of parameters and given observations (E-step), then, using these expectations the likelihood is maximized with respect to the parameters of the model (M-step).

$$\text{E-step:} \quad Q(\theta|\theta^t) = \sum_{A} \Pr(A|I, \theta^t) \ln \Pr(I, A|\,\theta) \tag{4}$$

$$\text{M-step:} \quad \theta^{t+1} = \arg\max_{\theta} Q(\theta|\theta^t) \tag{5}$$

[1] In principle there is a lowest-resolution factor $\Pr(A_L, I_L)$. We model this as $\prod_x \Pr(\mathbf{g}_L(x) \,|\, a_L(x)) \Pr(a_L(x))$. This is the $l = L$ factor of Equation 3, which should be read as having no quantities $\mathbf{f}_{L+1}$ or a_{L+1}.

where we have summarized all parameters of the model in θ, and θ^t represents the values of the parameters in the current iteration step t.

The main difficulty for our model lies in computing the expectations over the unknown labels. In this section only the resulting equations will be given. The derivation of the probability propagation in this hierarchical model are deferred to Appendix A.

Maximization We will start with the M-step by inserting (3) in (4),

$$Q(\theta|\,\theta^t) = \sum_A \Pr(A|I,\theta^t) \sum_{l=0}^{L} \sum_x \ln \Pr(\mathbf{g}_l(x), a_l(x)\,|\,\mathbf{f}_{l+1}(x), a_{l+1}(x), \theta) \tag{6}$$

$$= \sum_{l=0}^{L} \sum_x \sum_{a_l(x), a_{l+1}(x)} \Pr\big(a_l(x), a_{l+1}(x)|I,\theta^t\big) \ln \Pr\big(\mathbf{g}_l(x), a_l(x)|\,\mathbf{f}_{l+1}(x), a_{l+1}(x)\big) \tag{7}$$

Assuming we know the probability $\Pr(a_l(x), a_{l+1}(x)|I,\theta^t)$ for all parent/child label pairs, $a_l(x), a_{l+1}(x)$, we can search for the optimal parameters. At this point we have to commit to a parameterization of $\Pr(a_l(x)\,|\,a_{l+1}(x))$ and $\Pr(\mathbf{g}_l(x)\,|\,\mathbf{f}_{l+1}(x), a_l(x))$. We choose to use the same parameters for all positions so that we obtain homogeneous behavior across the image, but we will allow for different parameters at different layers. To keep $\Pr(a_l(x)\,|\,a_{l+1}(x))$ properly normalized we use

$$\Pr(a_l\,|\,a_{l+1}) = \frac{\pi_{a_l,a_{l+1}}}{\sum_{a_l} \pi_{a_l,a_{l+1}}} \tag{8}$$

We expected that the distribution of sub-sampled features conditioned on the features of the next layer is well described by a mixture of Gaussians with a linear dependency in the mean, and experiments have verified this. Our model represents a mixture where the label a selects the mixture component. We choose therefore a Gaussian distribution where the parameters are indexed by the labels and the dependency of the features is parameterized as a linear relationship in the mean.

$$\Pr(\mathbf{g}\,|\,\mathbf{f}, a) = \mathcal{N}(\mathbf{g}, M_a\mathbf{f} + \bar{\mathbf{g}}_a, \Lambda_a) \tag{9}$$

If the different features at a given pixel are orthogonal we expect that diagonal M and Λ will be sufficient. The parameter set is now $\theta = \{\pi_a, M_a, \bar{\mathbf{g}}_a, \Lambda_a | a = a_0, \ldots, a_L\}$. With the choices (9) and (8) the M-step is easy to solve. The maximum of (7) with respect to θ can be found by setting the derivatives with respect to the different parameters equal to zero and solving for the corresponding parameter. For $\pi^{t+1}_{a_l,a_{l+1}}$ we find

$$\frac{\pi^{t+1}_{a_l,a_{l+1}}}{\sum_{a'_l} \pi^{t+1}_{a'_l,a_{l+1}}} = \frac{\sum_x \Pr(a_l(x), a_{l+1}(x)|I,\theta^t)}{\sum_x \Pr(a_{l+1}(x)|I,\theta^t)}. \tag{10}$$

For the other update equations, let us denote the average over position at level l weighted by $\Pr(a_l(x)\,|\,I,\theta^t)$ by $\langle . \rangle_{t,a_l}$, i.e.,

$$\langle X \rangle_{t,a_l} = \frac{\sum_x \Pr(a_{l+1}(x)|I,\theta^t) X(x)}{\sum_x \Pr(a_{l+1}(x)|I,\theta^t)}. \tag{11}$$

Then the other update equations are

$$\bar{\mathbf{g}}_{a_l}^{t+1} = \langle \mathbf{g}_l \rangle_{t,a_l} - M_{a_l}^{t+1} \langle \mathbf{f}_{l+1} \rangle_{t,a_l} \tag{12}$$

$$M_{a_l}^{t+1} = \left(\langle \mathbf{g}_l \mathbf{f}_{l+1}^T \rangle_{t,a_l} - \bar{\mathbf{g}}_{a_l}^{t+1} \langle \mathbf{f}_{l+1}^T \rangle_{t,a_l} \right) \times \langle \mathbf{f}_{l+1} \mathbf{f}_{l+1}^T \rangle_{t,a_l}^{-1} \tag{13}$$

$$\Lambda_{a_l}^{t+1} = \left\langle \left(\mathbf{g}_l - M_{a_l}^{t+1} \mathbf{f}_{l+1} - \bar{\mathbf{g}}_{a_l}^{t+1} \right) \left(\mathbf{g}_l - M_{a_l}^{t+1} \mathbf{f}_{l+1} - \bar{\mathbf{g}}_{a_l}^{t+1} \right)^T \right\rangle_{t,a_l} \tag{14}$$

$$= \left\langle \left(\mathbf{g}_l - M_{a_l}^{t+1} \mathbf{f}_{l+1} \right) \left(\mathbf{g}_l - M_{a_l}^{t+1} \mathbf{f}_{l+1} \right)^T \right\rangle_{t,a_l} - \bar{\mathbf{g}}_{a_l}^{t+1} \bar{\mathbf{g}}_{a_l}^{t+1\,T}. \tag{15}$$

Since these expressions are mutually dependent, we must insert Equation 12 into Equation 13 and solve for $M_{a_l}^{t+1}$ to get

$$M_{a_l}^{t+1} = \left(\langle \mathbf{g}_l \mathbf{f}_{l+1}^T \rangle_{t,a_l} - \langle \mathbf{g}_l \rangle_{t,a_l} \langle \mathbf{f}_{l+1}^T \rangle_{t,a_l} \right) \left(\langle \mathbf{f}_{l+1} \mathbf{f}_{l+1}^T \rangle_{t,a_l} - \langle \mathbf{f}_{l+1} \rangle_{t,a_l} \langle \mathbf{f}_{l+1}^T \rangle_{t,a_l} \right)^{-1}. \tag{16}$$

So the update procedure is to compute $M_{a_l}^{t+1}$ using Equation 16, use this to compute $\bar{\mathbf{g}}_{a_l}^{t+1}$ according to Equation 12, and use both values in Equation 14 to compute $\Lambda_{a_l}^{t+1}$.

If we assumed diagonal M and Λ we can ignore the off-diagonal terms in these expressions. In fact, the component densities $\mathcal{N}(\mathbf{g}, M_a \mathbf{f} + \bar{\mathbf{g}}_a, \Lambda_a)$ factor into individual densities for each component of $\mathbf{g}$. We can replace Equations 16, 12, and 14 with their scalar versions and apply them to each component of $\mathbf{g}$ independently.

Expectation In the E-step we need to compute the probabilities $\Pr(a_l(x_l), a_{l+1}(x_l)|I, \theta^t)$ and $\Pr(a_l(x_l)|I, \theta^t)$ for given image data. But since these appear in both the numerator and denominator of all of the re-estimation equations, i.e. (10) and (11), we need these quantities only up to an overall factor. We can choose that factor to be $\Pr(I|\theta^t)$ and can therefore compute $\Pr(a_l(x_l), a_{l+1}(x_l), I|\theta^t)$ instead using

$$\begin{aligned} \Pr(a_l(x), a_{l+1}(x)|I, \theta^t) \Pr(I|\theta^t) &= \Pr(a_l(x), a_{l+1}(x), I|\theta^t) \\ &= \sum_{A \setminus a_l(x), a_{l+1}(x)} \Pr(I, A|\theta^t), \end{aligned} \tag{17}$$

and similarly for $\Pr(a_l(x_l)|I, \theta^t)$.

The complexity of computing these sums relates to the dependency structure of the variables A, which can be represented as a graph. From the literature on graphical models [7] we know that the cost of evaluating these sums grows exponentially with the clique sizes in that graph and only linearly with the number of cliques. If we choose the dependency such that every label is conditioned on only one label from the parent layer the clique size is minimal (Figure 5, right). For an image pyramid with sub-sampling-by-two that corresponds to a quad-tree structure. In a quad-tree a location x_l has only one parent $\mathsf{Par}(x_l)$ in layer $l+1$, and four children $\mathsf{Ch}(x_l)$ in layer $l-1$. If we do not restrict the dependencies, and maintain instead a structure in which every location in layer l is connected to every neighboring pixel in layers $l+1$ and $l-1$ (Figure 5, left), the entire label pyramid is one irreducible clique, and the exact evaluation of the sums becomes prohibitive.

Since we wish to compute the probability of hidden labels given the entire image pyramid, it will be necessary to propagate the probabilities of observations of the entire pyramid to a particular junction of label pairs. To do so the algorithm has to propagate the probabilities upwards, and then propagate them down again to the place of the particular label pair. On the way it is possible to marginalize (execute the sums) over the other labels. We will recursively define

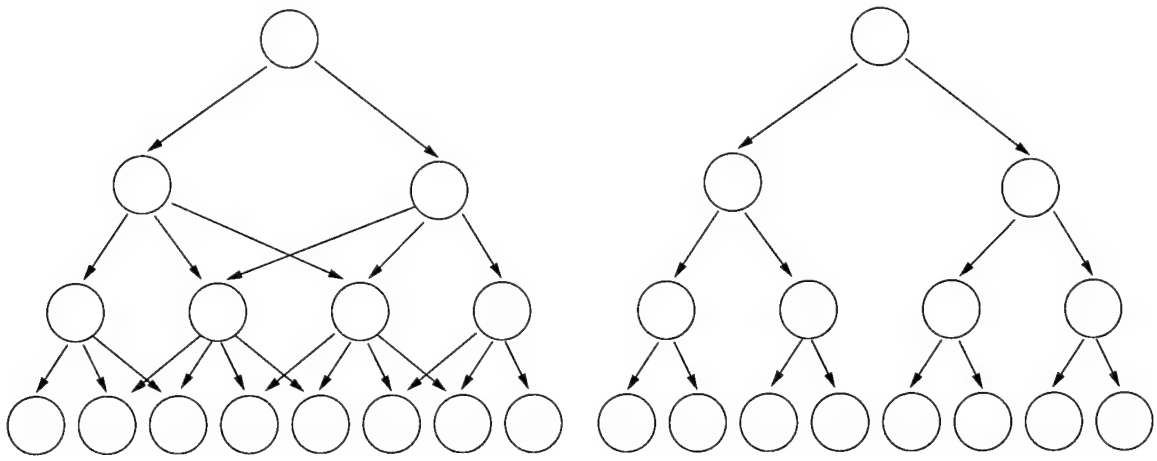

Figure 5: Dependency structure for the label pyramid. Left: dense graph where the smallest clique is the entire graph and probability computations increase exponentially with the tree size. Right: In a binary tree probabilities can be propagated efficiently. The disadvantage is that some neighboring nodes are very weakly linked, while others are very tightly linked.

quantities u and d representing the upwards and downwards propagating probabilities as evaluated in Appendix A,

$$u_l(a_l, x) = \Pr\big(\mathbf{g}_l(x)|\mathbf{f}_{l+1}(x), a_l\big) \prod_{x' \in \mathsf{Ch}(x)} \tilde{u}_{l-1}(a_l, x') \tag{18}$$

$$\tilde{u}_l(a_{l+1}, x) = \sum_{a_l} \Pr(a_l|\, a_{l+1}) u_l(a_l, x) \tag{19}$$

$$d_l(a_l, x) = \sum_{a_{l+1}} \Pr(a_l|\, a_{l+1}) \tilde{d}_l(a_{l+1}, x) \tag{20}$$

$$\tilde{d}_l(a_{l+1}, x) = \frac{u_{l+1}(a_{l+1}, \mathsf{Par}(x))}{u_l(a_{l+1}, x)}\, d_{l+1}\big(a_{l+1}, \mathsf{Par}(x)\big) \tag{21}$$

The upward recursion (18–19) is initialized at $l = 0$ with $u_0(a_0, x) = \Pr\big(\mathbf{g}(x)|\mathbf{f}_1(x), a_0\big)$ and ends at $l = L$. At layer L (19) reduces to $\tilde{u}_L(a_{L+1}, x) = \tilde{u}_L(x)$.[2] Since we do not model any further dependencies beyond layer L, the pixels at layer L are assumed independent. Considering the definition of u in (30) it is evident that the product of all $\tilde{u}_L(x)$ coincides with the total image probability,

$$\Pr(I|\theta^t) = \prod_{x \in I_L} \tilde{u}_L(x) = u_{L+1}. \tag{22}$$

The downward recursion (20–21) can be executed, starting with equation (21) at $l = L$ with $\tilde{d}_{L+1}(a_{L+1}, x) = \tilde{d}_{L+1}(x) = 1$.[2] The downwards recursion ends at $l = 0$ with equation (20).

With these quantities and using result (29) we can compute (17) as

$$\Pr(a_l(x), a_{l+1}(x), I|\theta^t) = u_l(a_l, x)\tilde{d}_l(a_{l+1}, x) \Pr(a_l|a_{l+1}) \tag{23}$$

$$\Pr(a_l(x), I|\theta^t) = u_l(a_l, x) d_l(a_l, x) \tag{24}$$

Obviously computations (18–24) in the E-step at iteration t need to be completed with fixed parameters θ^t.

[2]The (non-existent) label a_{L+1} can be thought of as a label with a single possible value. The conditional probability $\Pr(a_L|\, a_{L+1})$ turns into a prior $\Pr(a_L)$.

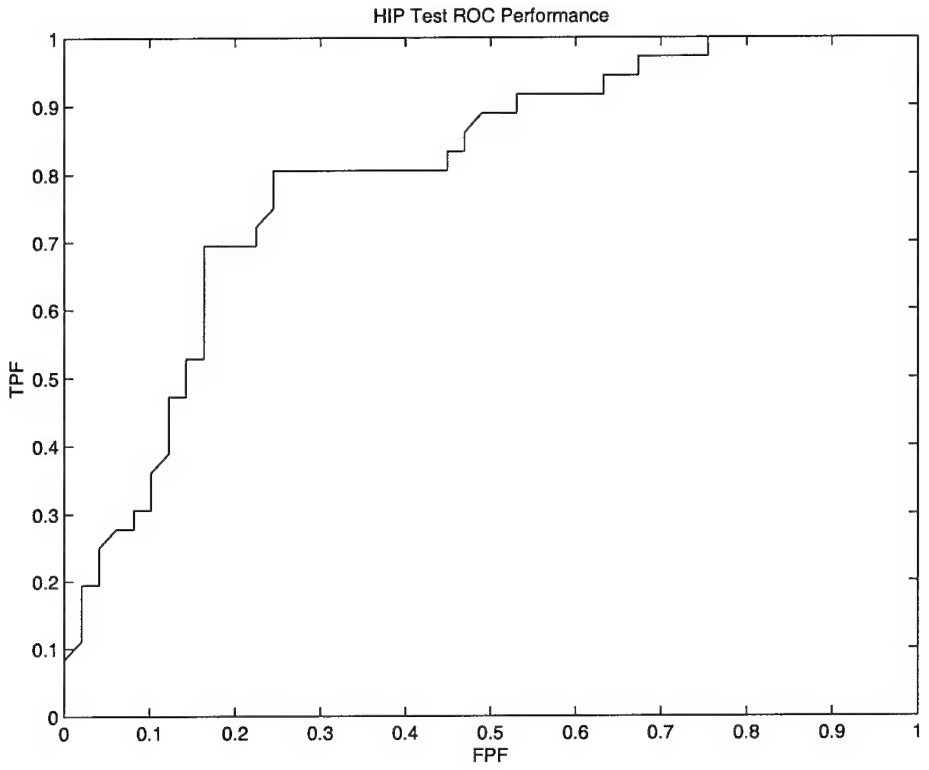

Figure 6: ROC curve for HIP model. Tested on 36 true positives and 50 false positives generated by The University of Chicago CAD system for mass detection.

2.3.3 Experiments

We have applied HIP to the problem of mass detection in mammographic CAD. We use the same data that was used to evaluate the HPNN (72 true positive and 100 false positive ROIs taken from the UofC CAD system). An ROC curve showing the HIP model's performance on the test data is shown in figure 6. For this particular model $A_z = 0.79$. A comparison between the HIP and HPNN performance on the same data is shown in table 2. Though the HIP model's performance on the test data was not as high as the HPNN, there was minimal model selection when training the HIP model, while there was significant model selection when training the HPNN (e.g. techniques such as varying the number of pyramid levels, nodes, etc and evaluating via cross validation). The fact that our initial application of the HIP model to mass detection gave reasonably good results is encouraging since we believe these results will be improved when we apply model selection techniques such as MDL.

3 HIP architecture selection with predictive MDL

As stated earlier, minimum description length techniques lend themselves well to HIP models because a description length of the images given the HIP model naturally encodes the compactness of the HIP distribution along with the likelihood of the data under the HIP distribution. MDL therefore gives us a natural means for making various architecture choices, e.g., the number of labels at each level in the hierarchy, the types of features to use, and so on.

In our first experiments we have chosen to use a *predictive MDL* or *PMDL* approach, due to its simplicity. We take

Sensitivity	Fine-to-Coarse HPNN Specificity	HIP Specificity
100 %	51 %	25 %
95 %	57 %	36 %
90 %	67 %	52 %
80 %	79 %	75 %

Table 2: Comparison of HPNN and HIP Detector Specificity (% reduction in false positive rate of UofC CAD system).

a training set $\mathcal{S}$, say all of the mass ROIs in the complete training set, and give the images within it some ordering. We then train the HIP model on the first n images in $\mathcal{S}$ and test it on the succeeding n images. The test results in a log-likelihood for these n test images, which we then add into a running sum R of these log-likelihoods. We then re-train on the first $2n$ images and test on the next n images. We repeat this until we have tested on the last images in $\mathcal{S}$.

To be precise, let $\mathcal{S}_k$ be the set of the first k elements from $\mathcal{S}$ (elements 1 through k) and let $\mathcal{S}_{j,k}$ be the set of elements in $\mathcal{S}$ from the j-th through the k-th, inclusive. We then follow this procedure:

1. Set $i = 1$ and $R = 0$.
2. Train (or re-train) the HIP model on $\mathcal{S}_{in}$.
3. Test trained HIP model on $\mathcal{S}_{in+1,in+n}$, adding the resulting log-likelihoods to R.
4. Let $i = i + 1$.
5. If $in >= |\mathcal{S}|$, exit, else return to step 2.

It has been shown [10] that the result R is asymptotically equal to the description length of the model and the data under the final trained model. Intuitively, one expects a U-shaped curve for R as a function of model complexity. A model that is too simple will give relatively poor results late in the PMDL procedure, since it can't adequately fit the data. A model that is too complex will give relatively poor results early in the PMDL procedure, since it over fits the data, and in fact overfits it for more iterations than a simpler model. Thus both too simple and too complex a model will have relatively high values for R.

Compared to leave-n-out cross-validation PMDL should be quite fast because it starts each re-training run at an architecture that was optimized on a large fraction of the new training set. One could do something similar with cross-validation, but in principle this adds bias. Besides the speed advantage, Rissanen claims that PMDL is more reliable than cross-validation [10].

We applied PMDL to choosing the number of components or labels at each level in HIP models for positive and negative ROIs.[3] The space we are searching, then, is the space of L-dimensional vectors $\mathbf{n}$ of natural numbers, i.e., $\mathbf{n} \in \mathbb{N}^L$. Unfortunately, this is a general nonlinear integer programming problem. Starting with an initial architecture $\mathbf{n}$, we search with the following procedure:

1. Compute or retrieve $R(\mathbf{n})$ for the current $\mathbf{n}$. (The R's for each architecture are stored to avoid re-computing them.)
2. For each l, $0 \leq l < L$, compute the l-th component g_l of a pseudo-gradient vector $\mathbf{g} \in \mathbb{R}^L$ as follows:
   (a) Compute $R(\mathbf{n}^{l+})$, where $\mathbf{n}^{l+}$ is the same as the current $\mathbf{n}$ except the l-the component has been increased by one.

[3] We did not include the determinants of the feature-extraction transformations in the log-likelihoods, but these are additive constants that are independent of the number of components in the HIP model. When we move on to choosing features we will need these determinants. We have computed these already using sparse matrix methods.

(b) If $n_l > 1$, compute $R(\mathbf{n}^{l-})$, where $\mathbf{n}^{l-}$ is the same as $\mathbf{n}$ except the l-the component has been decreased by one.

(c) Set g_l as follows:

$$g_l = \begin{cases} R(\mathbf{n}^{l+}) - R_i & \text{if } n_l > 1 \text{ and } R(\mathbf{n}^{l+}) > \max(R(\mathbf{n}^{l-}), R), \\ R_i - R(\mathbf{n}^{l-}) & \text{if } n_l > 1 \text{ and } R(\mathbf{n}^{l-}) > \max(R(\mathbf{n}^{l+}), R), \\ R(\mathbf{n}^{l+}) - R_i & \text{if } n_l = 1 \text{ and } R(\mathbf{n}^{l+}) > R, \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

3. If all components of $\mathbf{g}$ are 0, exit. We treat this as a local maximum.

4. Normalize $\mathbf{g}$ by dividing by the largest absolute value of any of its elements.

5. Search along the normalized $\mathbf{g}$. New architectures $\mathbf{n}'$ are tried, with $n'_l = n_l + \text{int}(dg_l)$, where d is a positive integer and int truncates toward zero. d is bounded so the components $n'_l \geq 1$ for all l. Set $\mathbf{n}$ to the new best architecture.

6. Return to step 1.

To classify an image, we can compute the ratio of the likelihoods (difference of the log-likelihoods) of the image under the two HIP models, that for the masses and that for the non-masses or false-positives of the CAD system that picked the ROIs. We then apply a threshold to the likelihood ratio to decide which should be detected. (This is equivalent to computing the probability of a mass being present given the image, and thresholding that.) Varying the threshold gives us an ROC curve.

So far the results of this procedure have been disappointing, but we have not yet performed a completely fair evaluation or comparison. An architecture for the model of the positive examples was started with one component per level. This finished with the architecture $\mathbf{n} = (17, 3, 3, 2, 1)$. (We had also bounded the number of components above at seventeen in order to speed the search and conserve memory.) Since the architecture search took so long, we started the search for a negative architecture with more components at each level, $\mathbf{n} = (8, 4, 2, 1, 1)$. This returned a best architecture of $\mathbf{n} = (8, 3, 2, 1, 1)$. The performance of this model pair on the test set was $A_z = 0.72$. By contrast, when we searched "by hand" for a best architecture for both models, we obtained the architecture $\mathbf{n} = (16, 11, 5, 2, 1)$ which gave the test-set performance $A_z = 0.79$. This later figure is biased since we were using the test performance as the search criterion. We are trying the biased procedure of starting the architecture search from the best architecture we found by hand, just to indicate how well PMDL correlates with the test set performance.

3.1 Comments on the architecture search procedure

Clearly our current search algorithm is sub-optimal. It can get stuck in local maxima, and in fact one might say it can get stuck at points that are not local maxima, since the points at which the algorithm exits are only better than those neighbors that differ in one component of the architecture vector. If changes in more than one component are allowed, many of these final points are probably not local maxima. Unfortunately, the number of neighbors that differ in more than one component is 2^L, and since training one architecture already takes several hours on a Sun Ultra 60 workstation, this more complete search takes a prohibitively long time.

Furthermore we frequently find architectures and layers for which both $R(\mathbf{n}^{l+})$ and $R(\mathbf{n}^{l-})$ are greater than R, yet we only search in one of the two directions, thus probably missing better local maxima part of the time.

One alternative to these heuristic approaches is exhaustive search, at least in a bounded region of the search space. This is optimal but very expensive. Unfortunately the unknown behavior of $R(\mathbf{n})$ means there are no better guaranteed-optimal methods.

It may be possible to develop a split-and-merge algorithm like that of Ueda, et al [13]. Such an algorithm would analyze the data conditioned on the model parameters, and attempt to decide which labels in the model could be

merged and which could be split. In this way a new architecture is always initialized at a relatively good fit to the data, rather than with random starting values. This precludes using PMDL to judge whether a particular split-and-merge operation improved the architecture, so we would have to use a conceptually more straightforward estimate of code length.

4 Key Research Accomplishments

1. Application of hierarchical pyramid neural network (HPNN) to mammographic mass detection. Results show a 51% reduction in false positive rate of The UofC CAD system for mass detection without loss in sensitivity.
2. Development of the hierarchical image probability (HIP) model for mammographic CAD. HIP is a generative model which allows for computing confidence measures based on the training data–an element which is often absent from CAD systems. More importantly, it's structure is well-suited for application of MDL model selection techniques. Initial application of HIP to mass detection shows that it can reduce the false positive rate of the UofC CAD system for mass detection by 25% without loss in sensitivity.
3. We have developed a search strategy and algorithm for applying predictive MDL to select a HIP architecture.

5 Reportable Outcomes

1. Disclosure/Patent Application "Hierarchical Image Probability Models", March 1999
2. Clay Spence, Lucas Parra and Paul Sajda, "Mammographic mass detection with a hierarchical image probability (HIP) model", submitted to SPIE Medical Imaging 2000.
3. Presentation "Hierarchical Pattern Recognition for Mammographic CAD", University of Pennsylvania, November 1998.

6 Conclusions

Under the first year of this project we have demonstrated the utility of hierarchical pattern recognizers for improving the performance of CAD systems for mass detection. Mass detection is currently the more difficult problem in mammographic CAD (compared to microcalcification detection). For CAD systems to gain clinical acceptance, false positives must be significantly reduced without loss in sensitivity. On a small research database, application of our HPNN model has resulted in a 51% reduction in false positive rate of the UofC CAD system for mass detection. However the HPNN models that we have trained are not well-suited to objective model selection techniques, such as MDL. Since objective model selection is often critical to maximizing the performance of a pattern recognizer, we developed a new hierarchical pattern recognition framework which we call the hierarchical image probability (HIP) model.

To justify the HIP framework, we have shown that image distributions can be exactly represented as products over pyramid levels of distributions of sub-sampled feature images conditioned on coarser-scale image information. We argued that hidden variables are needed to capture long-range dependencies while allowing us to further factor the distributions over position. As far as we know, no other approach captures these long-range dependencies, factors the distribution over positions (greatly simplifying the modeling problem), and gives true distributions of images. In our current model the hidden variables act as indices of mixture components. The resulting model is somewhat like a hidden Markov model on a tree.

Initial results of the HIP model for mammographic mass detection are slightly below the performance of the HPNN on the same data (HIP $A_z = 0.79$ vs HPNN $A_z = 0.85$). However, no significant model selection was done for the HIP model, while significant ad hoc model selection (e.g. architecture selection using cross validation error estimates)

was done for the HPNN.[4] Finally we have developed a search strategy and algorithm for applying predictive MDL (pMDL) and selecting the best label architecture for a HIP model. We are currently running experiments testing the HIP models constructed using pMDL.

6.1 "so what section"

Statistical pattern recognition is a key element in any mammographic computer-aided diagnosis system. Hierarchical pattern recognizers are particularly useful since they are capable of exploiting contextual and multi-resolution information for detecting clinically significant objects. Most statistical pattern recognizers that have been previously developed for mammographic CAD have been trained to estimate $\Pr(\text{class} \mid \text{image})$. By contrast HIP model, trained to estimate the probability distribution of images, $\Pr(\text{image})$, has many attractive features. One could use HIP for detection/classification in the usual way by training a distribution for each object class and using Bayes' rule to get $\Pr(\text{class} \mid \text{image}) = \Pr(\text{image} \mid \text{class}) \Pr(\text{class}) / \Pr(\text{image})$. We have reported our initial results for this application of HIP in this report.

Even though our original motivation for this model was to develop a framework for hierarchical pattern recognition which could exploit techniques in MDL model selection, there are other attractive features of the HIP framework which could have a major impact on the design and development of mammographic CAD systems. Since HIP computes $\Pr(\text{image} \mid \text{class})$, we could attempt to detect unusual images and reject them rather than trust the classifier; something that is not possible with models of $\Pr(\text{class} \mid \text{image})$. Building confidence measures into CAD systems is an open area of research and the HIP model provides a mechanism by which to generate these measures.

The HIP model has applications other than detection/classification. Since the HIP model is a generative model, one can use it to compress data, given the probability distribution of the objects of interest. If one wants lossless compression of a digital mammogram one need only train a HIP model for a set of mammographic images and then use the probability model to compress the data. More interesting is the application of HIP for lossy compression. In that case, one might train a HIP model on clinically significant objects, such as mammographic masses, since those are the parts of the image one would like to preserve–i.e. have minimal distortion and compression artifacts. The entire image can then be compressed using this model. Though there will be loss over regions of the mammogram which do not fit the model, those regions of clinical significance will be preserved since they will have a good fit to the probability model and require very few bits for compression.

[4] In addition the dataset used for training and testing was small (72 positives and 100 negative ROIs). Therefore additional data will be used to further test the difference between the two models.

References

[1] J. S. De Bonet, P. Viola, and J. W. Fisher III. Flexible histograms: A multiresolution target discrimination model. In E. G. Zelnio, editor, *Proceedings of SPIE*, volume 3370, 1998.

[2] Jeremy S. De Bonet and Paul Viola. Texture recognition using a non-parametric multi-scale statistical model. In *Conference on Computer Vision and Pattern Recognition*. IEEE, 1998.

[3] Robert W. Buccigrossi and Eero P. Simoncelli. Image compression via joint statistical characterization in the wavelet domain. Technical Report 414, U. Penn. GRASP Laboratory, 1998. Available at ftp://ftp.cis.upenn.edu/pub/eero/buccigrossi97.ps.gz.

[4] Rama Chellappa and S. Chatterjee. Classification of textures using Gaussian Markov random fields. *IEEE Trans. ASSP*, 33:959–963, 1985.

[5] Y. Le Cun, B. Boser, J. S. Denker, and D. Henderson. Handwritten digit recognition with a back-propagation network. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 396–404, 2929 Campus Drive, San Mateo, CA 94403, 1991. Morgan-Kaufmann Publishers.

[6] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. PAMI*, PAMI-6(6):194–207, November 1984.

[7] Michael I. Jordan, editor. *Learning in Graphical Models*, volume 89 of *NATO Science Series D: Behavioral and Brain Sciences*. Kluwer Academic, 1998.

[8] Mark R. Luettgen and Alan S. Willsky. Likelihood calculation for a class of multiscale stochastic models, with application to texture discrimination. *IEEE Trans. Image Proc.*, 4(2):194–207, 1995.

[9] J. A. Rissanen. A universal prior for integers and estimation of minimum description length. *Annals of Statistics*, 11(2):416–431, 1983.

[10] J. A. Rissanen. Information theory and neural nets. In Smolensky, Mozer, and Rumelhart, editors, *Mathematical Perspectives on Neural Networks*, pages 567–602, 1996.

[11] Paul Sajda, Clay D. Spence, John C. Pearson, and Robert M. Nishikawa. Exploiting context in mammograms: A hierarchical neural network for detecting microcalcifications. In Murray H. Loew and Kenneth M. Hanson, editors, *Medical Imaging 1996 — Image Processing*, volume 2710, pages 733–742, P.O. Box 10, Bellingham WA 98227-0010, 1996. SPIE.

[12] Clay D. Spence, Paul Sajda, and Robert M. Nishikawa. Dealing with uncertainty and error in truth data when training neural networks for computer-aided diagnosis applications. In Heinz U. Lemke, Michael W. Vannier, and Kiyonari Inamura, editors, *CAR '97: Proceedings of the 11th International Symposium on Computer Assisted Radiology and Surgery*, pages 352–357, Amsterdam, 1997. Elsevier.

[13] Naonori Ueda, Ryohei Nakano, Zoubin Ghahramani, and Geoffrey E. Hinton. SMEM algorithm for mixture models. In Michael S. Kearns, Sara A. Solla, and David A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 599–605, Massachusetts Institute of Technology, Cambridge, MA 02142, 1999. MIT Press.

[14] Song Chun Zhu, Ying Nian Wu, and David Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9(8):1627–1660, 1997.

A Belief propagation in HIP

Here we will show how to obtain the upwards and downwards propagation rules (18)-(21). All the computations can be executed locally. Consider the subgraph represented in Figure 7. Every node X can take on a discrete number of values. When we write $\sum_X$ we are referring to the sum over the those values. Assigned to every node is also an evidence node g_X, with a fixed value for given image data. When we write $g_X \ldots$ we are referring to g_X and all the evidence in the rest of the graph that can be reached through node X. In this notation the entire evidence provided by the image intensities I is the collection $\{g_A \ldots, g_B \ldots, g_C \ldots\}$. The probability required in the EM algorithm of Section 2.3.2 is

$$\begin{aligned}
\Pr(B, A, I) &= \Pr(B, A, g_A \ldots, g_B \ldots, g_C \ldots) && (26)\\
&= \Pr(A, g_A \ldots, g_C \ldots) \Pr(B, g_B \ldots | A) && (27)\\
&= \Pr(A, g_A \ldots, g_C \ldots) \Pr(B|A) \Pr(g_B \ldots | B) && (28)\\
&= d_B(A) \Pr(B|A) u(B) && (29)
\end{aligned}$$

In (27) we used the fact that conditioned on A the evidence coming through the children of A is independent from the rest of the tree beyond A. Since the children of A have in fact no other parent, all the probabilistic influence beyond that parent edge can be communicated only through A. Similarly in (28) we used the fact that the evidence g_B is independent from the children of B if conditioned on B. Finally in (29) we used the definitions for computing these probabilities recursively in an upwards and downwards probability propagation as follows:

$$\begin{aligned}
u(A) &\equiv \Pr(g_A, g_B \ldots, g_C \ldots | A) && (30)\\
&= \Pr(g_A|A) \Pr(g_B \ldots | A) \Pr(g_C \ldots | A) && (31)\\
&= \Pr(g_A|A) u_B(A) u_C(A) = \Pr(g_A|A) \prod_{X \in \mathrm{Ch}(A)} u_X(A) && (32)\\
u_B(A) &\equiv \Pr(g_B \ldots | A) && (33)\\
&= \sum_B \Pr(B|A) \Pr(g_B \ldots | B) && (34)\\
&= \sum_B \Pr(B|A) u(B) && (35)
\end{aligned}$$

We have used in (31) and (34) conditional independence when conditioning on A and B respectively. In (35) we have used definition (30) for node B and in (32) we used definition (33) for the children of A. The downward propagating probability is defined and computed as,

$$\begin{aligned}
d_B(A) &= \Pr(A, g_A \ldots, g_C \ldots) && (36)\\
&= \Pr(g_C \ldots | A) \Pr(A, g_A \ldots) && (37)\\
&= \frac{u(A)}{u_B(A)} d(A) && (38)\\
d(B) &\equiv \Pr(B, g_A \ldots, g_C \ldots) && (39)\\
&= \sum_A \Pr(B|A) \Pr(A, g_A \ldots, g_C \ldots) && (40)\\
&= \sum_A \Pr(B|A) d_B(A) && (41)
\end{aligned}$$

Again, we have used the conditional independences when conditioning on A in (37), (38), and (40). One can verity (38) by inserting the corresponding definitions and canceling the factor $\Pr(g_A|A)$ to recover (37).

Equations (18–21) in Section 2.3.2 just rewrite this upwards and downwards propagation. Corresponding to the definitions in this Appendix, the propagation probabilities u and d of Section 2.3.2 are a function of the label values. The notation differs there only in that the labels have to be identified by their location in the label pyramid.

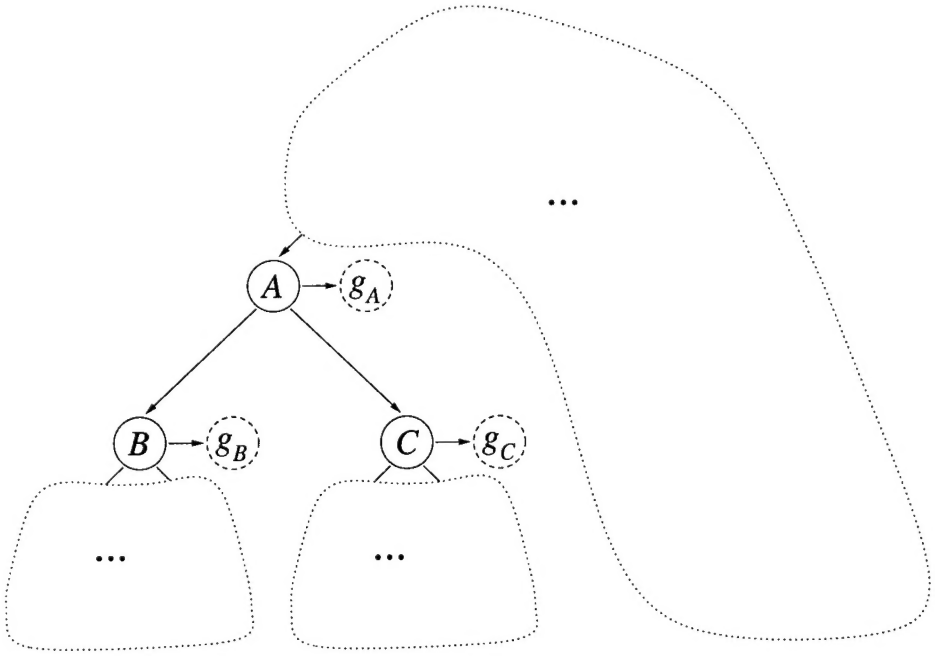

Figure 7: Subgraph of the label pyramid. Conditioned on A the variables that are connected to A become independent, such as labels B, C, and the evidence node g_A. These variables are also conditionally independent to the joint variables that can be reached going upwards to the rest of the tree structure.